Using of Additional Packages of Components for Accelerated Application Development

Assistant Prof. PhD Stoicho Stoev
University of Economics - Varna, Varna, Bulgaria
s.stoev@ue-varna.bg

Abstract

The application of information technology is embracing a growing part of modern business. This necessitates on the one hand the development of increasingly diverse information systems and the rapid development of products for small and medium-sized businesses. An appropriate approach is to use additional packages for development environments. They help optimize the process of building one or a range of applications with similar functionality. The add-on packages extend the capabilities of the environment by either offering standard capabilities or offering entirely new ones. For some additional packages, the internal integration of the individual components is implemented, which significantly facilitates and accelerates the development of a single application. In the presentation we will present the ideology and principles of using such packages to assist developers.

Keywords: information systems, development, software

JEL Code: C88; doi:10.36997/IJUSV-ESS/2019.8.2.171

Въведение

Развитието на съвременният бизнес е свързано с все по-широко приложение на информационните технологии. Еволюционно, използването на информационни системи автоматизира първоначално основните бизнес-процеси, като счетоводни, производствени, логистични и др. За решаване на тези задачи, традиционно фирмите използват готови решения, доказали се в практиката. С натрупване на практически опит в използване на информационни системи, се появява тенденция и в автоматизиране на второстепенни процеси с цел повишаване на ефективността на организацията. Възниква необходимост от реализиране на сравнително малки приложения за решаване на специфични задачи. От гледна точка на разработчиците на такива системи, от съществено значение е проектирането и реализиране им да се характеризира с изразходване на минимални ресурси. От съществено значение е използваната развойна среда, база от данни и/или наличието на подходящи инструменти за ускоряване на процесите по реализиране на малки информационни системи. В този смисъл, целта на изследването е да предложи някои решения за по-бързо разработване на приложения. Основно внимание ще се обърне на възможността за използване на допълнителни пакети към развойната среда за ускоряване и улесняване при реализиране на малки или средни информационни системи.

1. Област на приложение на допълнителни пакети компоненти. Вид приложения

Използване на информационни системи за автоматизиране на специфични бизнес задачи е свързано с конкретизиране на техният вид и област на приложения. Те може да се охарактеризират като:

-*Десктоп приложения*. Реализиране на система, която обслужва едно или две работни места предполага изграждане на десктоп приложения. Изборът на Уеб-решение в случая е неефективно поради допълнителните изисквания за интернет инфраструктура. В някои случаи, работните места са отдалечени и неразполагат с интернет връзка. Следователно, десктоп системите инфраструктурно са по-добро решение, изискващо по-малко ресурси.

-*Уникалност*. В много от случаите, реализираните информационни системи предназначени за решаване на конкретни задачи са уникални. Тоест при тях трудно се осъществява мултиплициране на клиентите, без допълнителното преработване или

адаптиране на приложението. Това принуждава фирмата-разработчик да цели по-бързото им разработване, без разходване на много ресурси.

Следователно информационните системи, за които в процеса на разработка използването на допълнителни пакети от компоненти е удачно, са сравнително малки, изграждани с ускорени темпове, в екип от един или двама програмисти.

Използвани среди и езици.

При разработване на десктоп приложения изборът на програмен език и развойна среда са от съществено значение. На съвременният етап от развитието на компютърните технологии съществува богата гама от езици за програмиране. Една част от тях са нововъзникнали, включващи последните нововъведения в областта. Друга част макар и по-стари, като реализация се развиват доколкото позволява идеологията им.

Изборът на подходящ програмен език може да обобщим в няколко критерия:

-*Възможности на езика*. Въпреки, че в по-голямата си част съвременните езици за програмиране имат сходни възможности, то няколко характеристики са значими: процедурни, обектно-ориентирани, визуални, бързодействие и др.

-*Възможности на развойната среда*. За изграждане на едно приложение, освен възможностите на езика за програмиране, голямо влияние оказва и интегрираната среда за разработка. Производителността значително се подобрява, ако средата е визуална, с интегрирани инструменти за дебъгинг, структуриране на кода, включване на допълнителни инструменти и др.

-*Достъп до база от данни*. Всяка информационна система зависи от използването на определена база от данни. Вариантите за достъп до данните предопределя както времето за разработка на приложенията, така и тяхното бързодействие. Някои среди предоставят натив (директни) контроли за връзка с определени база от данни, което значително повишава бързодействието. Други разчитат основно на ресурсите предоставени от операционната система.

-*Обектноориентирана идеология*. Не всички програмни езици се характеризират с цялостна обектноориентирана технология. Развитието на по-стари езици за програмиране достига само до обектно базирани компоненти или въобще липса на реализиране на ООП. Това довежда до проблеми при разширяване на възможностите на езика и/или до по-пълноценно използване на съвременните, обектноориентирани операционни системи.

Изборът на програмен език и развойна среда са от съществено значение за функционалността на изграждените приложения и необходимото време за тяхното създаване. Популярността на определен език предполага наличието на повече решения и информация за него, налична в интернет.

Популярният TIOBE Index, класацията за програмни езици Redmonk и анкета на Stack Overflow разкриват кои са програмните езици класирани по своята популярност[1] (Таблица 1). Рейтингът на програмните езици се формира въз основа на данните от търсенията в Google, YouTube, Baidu, Amazon, Wikipedia и други източници в мрежата.

Резултатите в таблицата отразяват популярността на езиците за програмиране в глобален мащаб. За разработката на десктоп-приложения в рамките на България, може да откроим няколко от изброените езици: C++, C#, Delphi/Object Pascal, Python и Visual Basic. Популярността на тези езици в софтуерната индустрия, може да се обясни както с развитието на учебните програми в образованието, така и с периодите на възникване на някои от компаниите и паралелното развитие на конкретни езици за програмиране. От гледна точка на средите за разработка предпочитанията са насочени към Microsoft Visual Studio и Delphi/ Embarcadero. И двете среди предлагат отворена архитектура към разширяване на съществуващите инструменти и инсталиране на допълнителни пакети от компоненти.

[1] Източник - https://www.economy.bg/home/view/21602/Koi-sa-naj-populyarnite-programni-ezici-

Jan 2016	Jan 2015	Change	Programming Language	Ratings	Change
1	2	^	Java	21.465%	+5.94%
2	1	v	C	16.036%	-0.67%
3	4	^	C++	6.914%	+0.21%
4	5	^	C#	4.707%	-0.34%
5	8	^	Python	3.854%	+1.24%
6	6		PHP	2.706%	-1.08%
7	16	^^	Visual Basic .NET	2.582%	+1.51%
8	7	v	JavaScript	2.565%	-0.71%
9	14	^^	Assembly language	2.095%	+0.92%
10	15	^^	Ruby	2.047%	+0.92%
11	9	v	Perl	1.841%	-0.42%
12	20	^^	Delphi/Object Pascal	1.786%	+0.95%
13	17	^^	Visual Basic	1.684%	+0.61%
14	25	^^	Swift	1.363%	+0.62%
15	11	vv	MATLAB	1.228%	-0.16%
16	30	^^	Pascal	1.194%	+0.52%
17	82	^^	Groovy	1.182%	+1.07%
18	3	vv	Objective-C	1.074%	-5.88%
19	18	v	R	1.054%	+0.01%
20	10	vv	PL/SQL	1.016%	-1.00%

Таблица 1. Класиране на най-популярните езици за програмиране за 2016 г.

2. Възможности за използване на допълнителни пакети от компоненти.

Анализът на развойните среди за програмиране показва, че за разработчиците се предлагат основни контроли, базисни за операционните системи. Следователно при разработване на приложения, се налага основни функционалности да се дописват или да се избере вариант за закупуване на допълнителни пакети от контроли.

Като недостатъци на базисните елементи на интерфейса част от развойната среда може да се обобщят в няколко функционалности:

-***Липса на вграден контрол на входящият поток от данни.*** Основните контроли предлагат възможности за валидизация на въвежданите данни, но за този контрол се разчита на разработчиците. Това значително удължава времето за изграждане на едно приложение, защото е необходимо да се мултиплицира програмен код за сходни операции по контрол на входните данни. Още повече, че при някои типове приложения се разчита на типизирана информация. Такива примери може да се посочат за изцяло числови данни като телефони, ЕГН, количества, стойност и др.

-***Визуализиране на съвременни данни.*** Развитието на информационните технологии предполага включване в съвременните информационни системи на данни като изображения, интернет връзки, документи, диаграми, бизнес-графики, работни планове и др. Особено при диаграмите и бизнес-графиките развойните среди предлагат силно ограничени възможности.

Реализиране на визуализацията на такъв тип информация чрез базисните контроли на средата е трудоемък процес.

-***Интегриране на различни контроли в общ интерфейс.*** Основните развойни среди предлагат самостоятелни контроли без възможност за връзка между тях. За осъществяване на такава интеграция се разчита единствено на разработчиците. Реализирането на такова обединение на различни контроли в един е свързано с квалифициран и трудоемък процес на програмиране. Примери за такава интеграция може да се посочи въвеждане на числови данни с вграден калкулатор, шаблонизатор за въвеждане на ЕГН и т.н.

-***Ограничени възможности за таблична информация.*** При изграждане на информационни системи се разчита на използване на таблични данни. В развойните среди за тази цел се използват Grid-контролите. Като базисни елементи на интерфейсът те са силно ограничени откъм функционалности. При тях липсват основни възможности като:

1. *Сортиране на данните*. Сортиране по една или няколко колони в таблицата може да реализира единствено програмно. В следствие на това, за всяка таблица използвана в потребителският интерфейс се налага дописване на необходимия код. Това води или до ограничаване на удобството за потребители на ИС или до утежняване на разработването на проекта.
2. *Филтриране на данните*. Филтрирането на информацията е основна функционалност в съвременния потребителски интерфейс. Възможността да изолираш информацията до данни необходими за потребителя е задължителна функционалност за всяка информационна система. За съжаление за тази възможност, отново се разчита на разработчика, без да са предвидени варианти за автоматизиране на този процес.
3. *Таблично въвеждане на данни*. Друг проблем при използване на базисните Grid-контроли е въвеждане на данни директно в табличен вид. Обикновено практиката налага въвеждането на данни в отделни колони в таблицата да се осъществява, чрез различни контроли. Тоест в някои колони е възможно да се предпочита избор от краен вариант данни, в други чрез контроли за дати, в трети стойност по определен шаблон и др. Без използване на разширени допълнителни контроли този процес е трудоемък.
4. *Функционалност Master-Detail*. Тази функционалност се изразява във вграждане на една таблична форма в друга. Това позволява на потребителят, бърз и лесен достъп до допълнителна информация, свързана с конкретни данни. Основните контроли в развойните среди не предвиждат възможност за реализиране на такава функционалност.
5. *Функционалност Pivot*. Pivot функционалността позволява групиране на информацията в табличната форма по една или повече колони. Тя позволява на потребителя по свое усмотрение да обобщава резултатни данни. Това разширява възможностите на информационните системи, като ги прави по-гъвкави и по-малко зависими от разработчиците, в следствие на автоматизиране на процеса по обработката на информацията.

От изложените проблеми при използване на базисните контроли в развойните среди се налага изводът, че фирма-разработчик на информационни системи трябва да търси начини за разширяване на възможностите на средата. Възможните решения не са много и зависят от ресурсите, с които тя разполага. Вариантите за разширение на потребителския интерфейс може да се обобщят в три направления:

1. **Закупуване на готови пакети от контроли**, предлагани от други фирми, специализирали се в областта. Този подход се предоставя две основни предимства. На първо място разработчика разполага с професионално създадени пакети от компоненти, в следствие на специализацията на фирмата производител. Пакетите от контроли са функционални,

производителни и предлагат интеграция между отделните елементи. От важно значение и постоянната поддръжка и ъпдейт на продуктите. На второ място е финансовият аспект. Стойността на такъв пакет е значително по-ниска от варианта, фирмата да финансира собствена разработка, като отдели персонал и техника за продължителен период от време. Като недостатък може да се посочи универсалността на продуктите, поради което понякога е необходимо да се допълват с доработка за решаване на специфични задачи.

2. **Разработване на собствени решения**. При изграждане на големи проекти и при фирмите с традиции в областта на софтуерната индустрия с течение на времето се изграждат собствени библиотеки от разширения на развойната среда. Тези допълнения са в отговор на развойната технология на фирмата, изискванията на продуктите, които развиват или от липсата на цялостно или частично готови външни решения. Като цяло този подход не е предварително планиран, а в следствие на постепенното натрупване на различни решения през периода на създаване на отделни продукти. Поради това, той рядко се среща в чистия си вид.

3. **Смесен подход**. Най-често фирмите използват смесен подход, при който се закупуват външни пакети от контроли и паралелно се натрупват потребителски библиотеки със собствени решения. Този подход се наложил в практиката поради две основни предимства. На първо място инвестирайки във външен продукт, фирмата има възможност веднага да се съсредоточи в разработката на търговски софтуер. Не се налага да се отделят време, персонал и финансови ресурси за създаване на собствени пакети от контроли. На второ място собствените библиотеки са тясно специализирани, постепенно изграждани (т.е натрупва се само този код, който се е доказал) и фирмата има пълен достъп до кода.

Изборът за създаване или закупуване на допълнителни пакети от компоненти зависи от мащаба на фирмата разработчик, от вида и структурата на изграждените приложения, както и възможностите на развойната среда. Независимо кой от изброените подходи се избере от компанията, трябва да се отбележи, че при разширяване на фирмата, новият персонал се налага да се обучи за работа с допълнителният инструментариум.

Анализ на комерсиални пакети от контроли.

Както беше отбелязано за изграждане на търговски софтуер в България, най-популярните среди за разработка се явяват Microsoft Visual Studio и Delphi/ Embarcadero. В таблица 2 са представени най-предпочитаните пакети от компоненти, според сайта componentsource.com.

Visual Studio	Embarcadero
Janus GridEX for .NET	ExpressQuantumPack
Janus Systems	DevExpress
DevExpress DXperience	TeeGrid for VCL/FMX
DevExpress	Steema Software
ComponentOne Studio Enterprise	Inspex
GrapeCity	Raize Software
Infragistics Professional	RAD Solution Pack
Infragistics	Embarcadero

Таблица 2. Най-популярни пакети от контроли[2]

[2] Източник - https://www.componentsource.com/

От данните в таблицата може да е направи изводът, че и за двете развойни среди на водещи позиции се класират продуктите на фирмата DevExpress[3].

Предимствата на продуктите на фирмата са:

- Интегрираност на отделните компоненти в цялостен пакет.
- Разширена функционалност на предлаганите контроли.
- Реализиране на функционалности чрез дизайн, тоест без допълнителен код.

Основният пакет от контроли съдържа няколко групи елементи:

-**Data Editors.** Предоставя разширени контроли за изграждане на потребителски интерфейс, аналогични на тези от развойната среда, но с допълнителни функционалности. Те са във вариант както за директно въвеждане на данни, така и чрез извличане от база от данни.

- **Navigation.** Съдържат усъвършенствани контроли за навигация и групиране на потребителският интерфейс. Може да се избира между менюта, барове, панели и др. със задаване на собствен изглед.

-**Analytics & Reporting.** Предлага атрактивни бизнес-графики, реализиране на визуални географски карти, пивот таблици и др.

-**Office Inspired.** Съдържа контроли за обработка на таблични данни, експорт и импорт от документи на Microsoft Office, експорт към PDF файлове, създаване на работни графици и др.

-**Layout & Windows UI.** Предоставя автоматичното подреждане на контролите за потребителският интерфейс, като пропорционално преоразмеряване, промяна на изгледа им (skins, цветове, шрифтове и др.), което допринася за адаптиране на приложението към желанията на потребителя.

-**Dialogs & Notifications.** Компонентите в този раздел заместват традиционните диалогови прозорци и кутии за съобщения, които не поддържат DevExpress skins. Заменяйки тези обекти по подразбиране от развойната среда с аналогичните на DevExpress, може да се постигне унифициран вид в цялото приложение.

-**Printing system.** Фирмата предлага собствена разработка за извеждане на печат. Предимства на системата са автоматизирания процес, при който желаната информация от потребителския интерфейс се подава към принтиращото устройство.

-**ExpressQuantumPack.** ExpressQuantumGrid на DevExpress е един от най-добрите компоненти за редактиране / оформяне на данни и се доставя с множество функции с висока ефективност, така че да може лесно да се управлява необходимата информация, както диктуват бизнес потребностите.

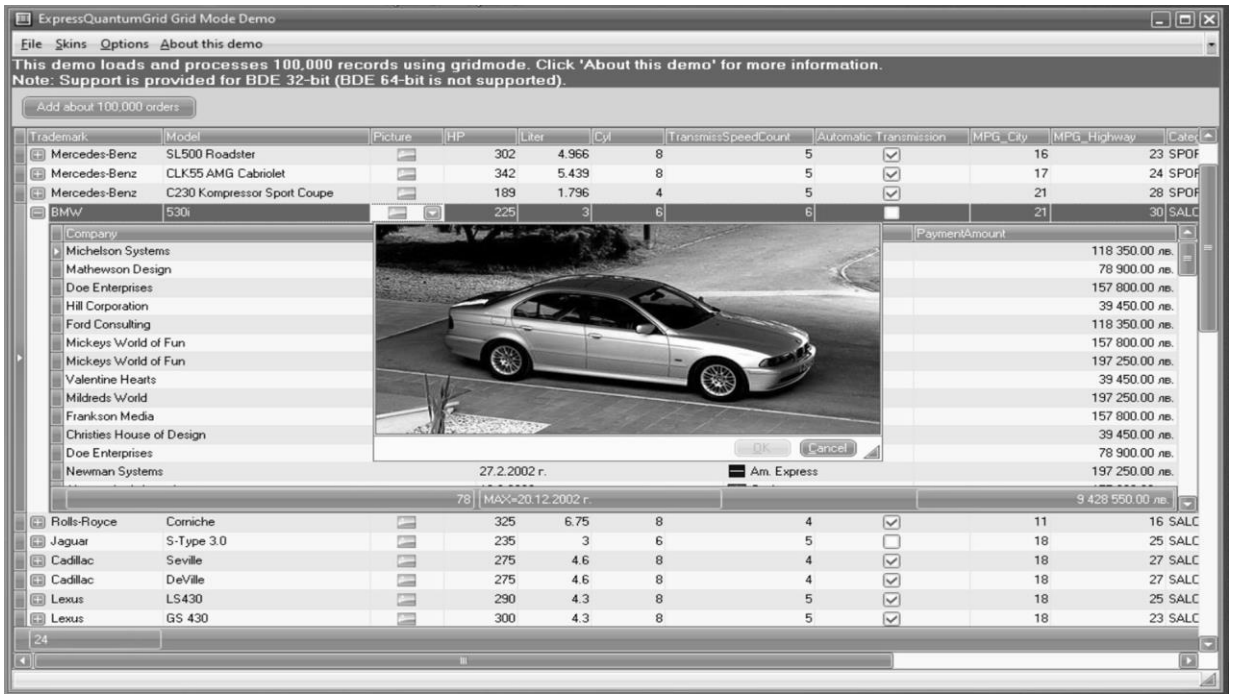
За популярността на продуктите обяснява и фактът, че те се предлагат и чрез сайта на Microsoft за разработчици на Visual Studio. За качеството и функционалността на пакетите, оценката се изразява в 18 награди за най-добър инструмент в областта[4].

3. Някои приложения на пакети с компоненти на фирмата DevExpress.

Пакетите от компоненти на фирмата DevExpress предлагат мощен инструмент за изграждане на съвременни десктоп приложения. Те позволяват автоматизиране на основни процеси по реализиране на потребителски интерфейс. Основно предимство на продукта е интегрираността на отделните контроли в една обща система. Това означава, че те могат да се използват както самостоятелно, така и като част от други контроли.

Основен за пакета се явява контролът QuantumGrid, предназначен за визуализиране на таблични данни. Негов примерен изглед е представен на фиг.1.

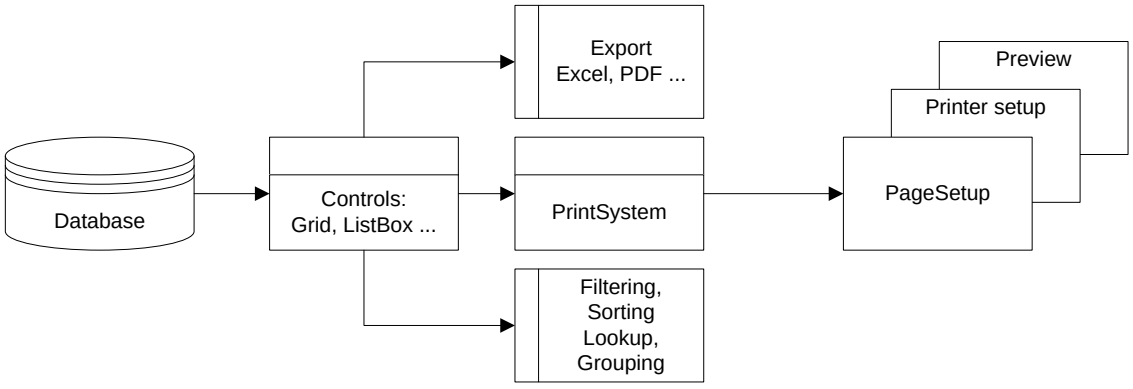
[3] https://www.devexpress.com/
[4] Източник - https://www.devexpress.com/



Фиг.1 Използване на контроли за визуализиране на таблични данни.

QuantumGrid значително ускорява разработката на приложения, защото процесът по дизайн на изгледа на данните не се осъществява програмно, а изцяло е организиран чрез настройка на свойствата му. В този смисъл, дейността на разработчика е да подаде необходимите данни от съответна SQL-платформа и да настрои необходимите характеристики на грида, като формат на данните, контроли за визуализиране на специфични данни, връзка от типа Master-Detail, обобщаващи данни и др. Потребителят има възможност да избере начина на сортиране на редовете, евентуалното им филтриране по собствени критерии, групирането по една или няколко колони и др.

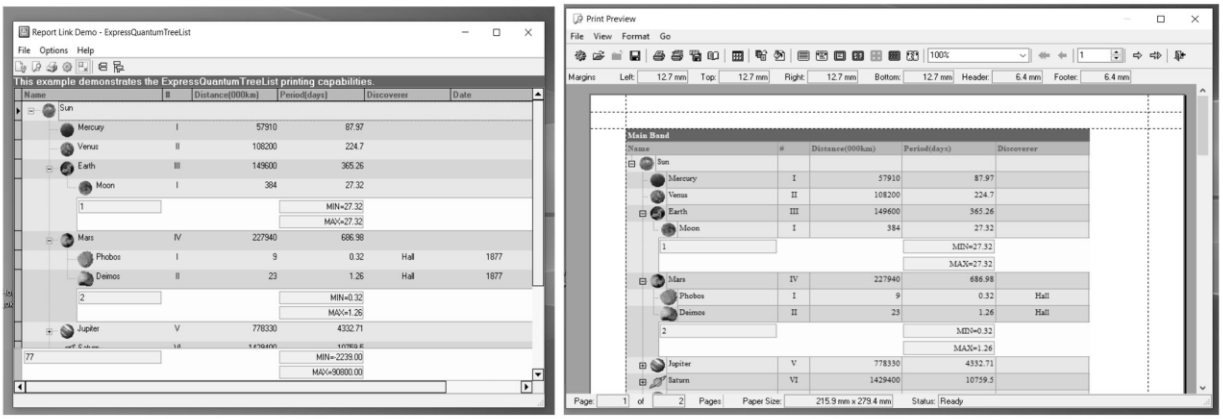
Изграждането на потребителски интерфейс, чрез използване на различни контроли се осъществява чрез модела показан на фиг.2. Прилагането на този модел е възможно да се реализират множество екрани от приложението с минимализиране на допълнителен програмен код.



Фиг.2 Модел за интегриране на отделни компоненти

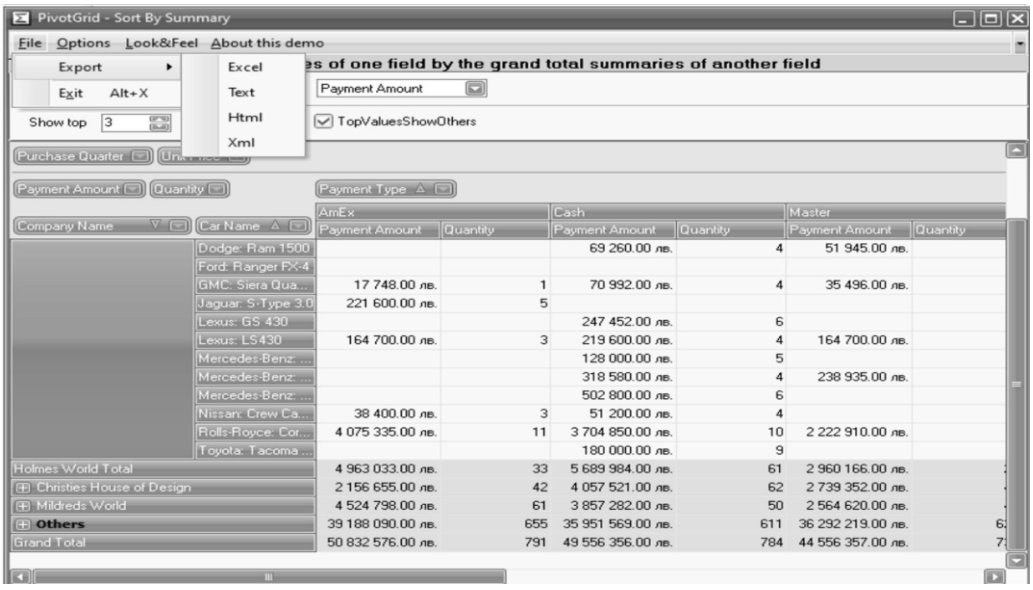
Особено полезно е прилагането на модела при изграждане на справочна информация за информационната система. Основната задача на програмистът се явява извличане на необходимата информация от базата данни и свързването и с необходимите контроли. В

процеса на извеждане на информацията към изходно устройство е показателен за гъвкавостта на системата. От гледна точка на разработчика, е нужно само да свърже контролите на PrintSystem и да укаже от кой конкретен контрол ще се печата информацията. От гледна точка на потребителят, той има възможност да промени формата на принтираната информация по негово усмотрение, а не във варианта зададен от програмиста. Пример за използване на връзката между PrintSystem и QuantumGrid е показана на фиг.3.



Фиг.3 Интегрирана система за принтиране на информация.

Разширение на възможностите на QuantumGrid се явява контролът PivotGrid (фиг.4). Освен функционалностите на основният грид-контрол, предимствата му се изразяват най-силно при визуализиране на кръстосани данни. Чрез тях потребителят им възможност за по-добър преглед на обобщена информация. PivotGrid-ът преобразува обикновени таблични данни в кръстосани, чрез групиране на определени колони.



Фиг.4 Визуализиране на кръстосана информация

Пакетите от компоненти на фирмата DevExpress за развойните среди Microsoft Visual Studio и Delphi/ Embarcadero, както ускорява разработване на потребителски приложения, така и значително разширява възможностите на средата. Тяхното използване е ефикасно не само при изграждане на големи информационни системи, но са приложими при създаване на малки специализирани решения.

Заключение

В заключение може да се обобщят няколко извода. На първо място базисните среди за програмиране не предлагат възможности за изграждане на модерни, информационни системи, без допълнителното разширяване на предлаганите контроли по интерфейса. Това предполага разходване на ресурси за компенсиране на тези недостатъци. На второ място закупуване и използване на допълнителни пакети от компоненти, значително подобрява потребителският интерфейс и ускорява разработката нови или разширяването на съществуващи приложения. Още повече, че разработчиците на пакети притежават по-голяма квалификация и знания свързани, както с развойната среда, така и с операционната система. Това позволява повишаване на качеството на крайният продукт. В следствие повечето софтуерни фирми използват допълнителни пакети от компоненти за усъвършенстване на своите продукти.

References

1. Andreou, A. and Tziakouris, M. (2007) A quality framework for developing and evaluating original software components. Information and Software Technology. 49 (2), pp 122-141
2. Bertoa, M., Troya, J. and Vallecillo, A. (2006) Measuring the usability of software components. Journal of Systems and Software. 79 (3), pp 427-439
3. Carmel, E. and Sawyer, S. (1998) Packaged software development teams: what makes them different?. *Information Technology & People*, 11 (1), pp. 7-19
4. Chinnaiyan, R. and Somasundaram, S. (2010) Evaluating the reliability of component-based software systems. *International Journal of Quality & Reliability Management*, 27 (1), pp. 78-88
5. Dubé, L. (1998) Teams in packaged software development. *Information Technology & People*, 11 (1), pp. 36-61
6. Gaedke, M., Meinecke, J. and Nussbaumer, M. (2005) Aspects of service-oriented component procurement in web-based information systems. International Journal of Web Information Systems. 1 (1), pp. 15-24
7. Hofmann, H., Muench, V. and Stynes, J. (1999) Mechanisms of component-oriented software development. *Internet Research*, 9 (1), pp. 66-75
8. Kimmel, P., Bucknall, J, and Kunk, J. (2010) Professional DevExpress™ ASP.NET Controls. Wiley Publishing, Inc.
9. Koi sa naj-populqrnite programni ezici? [Online] Available from: https://www.economy.bg/home/view/21602/Koi-sa-naj-populyarnite-programni-ezici- [Accessed 10/10/2019]
10. Koziolek, H. (2010) Performance evaluation of component-based software systems: A survey. Performance Evaluation. 67 (8), pp 634-658
11. McBride, T., Henderson-Sellers, B. and Zowghi, D. (2007) Software development as a design or a production project. *Journal of Enterprise Information Management,* 20 (1), pp. 70-82
12. Panayotova, G., Dimitrov, G.P., Petrov, P., & Os, B. (2016) Modeling and data processing of information systems. 3rd International Conference on Artificial Intelligence and Pattern Recognition, AIPR 2016, pp.154-158.
13. Shatnawi, A., Seriai, A., Sahraoui, H. & Alshara, Z. (2017) Reverse engineering reusable software components from object-oriented APIs. Journal of Systems and Software. 131, pp 442-460
14. Tamura, Y. , Yamada, S. and Kimura, M. (2006) Software reliability modeling in distributed development environment. *Journal of Quality in Maintenance Engineering,* 12 (4), pp. 425-432
15. Tang, J., Mu, L., Kwong, C. and Luo, X. (2011) An optimization model for software component selection under multiple applications development. European Journal of Operational Research, 212 (2), pp 301-311
16. Wegner, H., Hupe, P. and Matthes, F. (2000) A process-oriented and content-based perspective on software components. Information Systems, 25 (2), pp 135-156
17. Wijayasiriwardhane, T. and Lai, R. (2010) Component Point: A system-level size measure for Component-Based Software Systems. Journal of Systems and Software, 83 (12), pp 2456-2470